PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : H04L 12/56, H04Q 11/04

A1

(11) International Publication Number: **WO 99/35792**

(43) International Publication Date: 15 July 1999 (15.07.99)

(21) International Application Number: PCT/US99/00684

(22) International Filing Date: 12 January 1999 (12.01.99)

(30) Priority Data:
09/005,740 12 January 1998 (12.01.98) US

(71) Applicant: CABLETRON SYSTEMS, INC. [US/US]; 35 Industrial Way, Rochester, NH 03867 (US).

(72) Inventors: CHARNY, Anna; 408 Dutton Road, Sudbury, MA 01776 (US). KRISHNA, Pattabhiraman; 21 Royal Crest Drive #9, Marlboro, MA 01752 (US). PATEL, Naimish; 32 Monteiro Way, N. Andover, MA 01845 (US). SIMCOE, Robert, J.; 11 Brookway Road, Westboro, MA 01581 (US).

(74) Agent: SORKIN, Paul, D.; Wolf, Greenfield & Sacks, P.C., 600 Atlantic Avenue, Boston, MA 02210 (US).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published

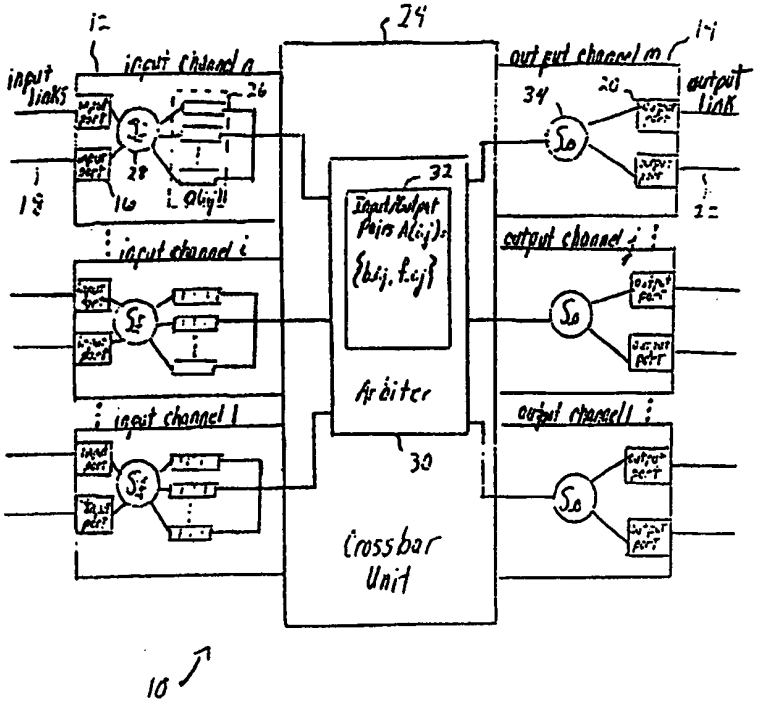
With international search report.

Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: METHOD FOR PROVIDING DELAYS INDEPENDENT OF SWITCH SIZE IN A CROSSBAR SWITCH WITH SPEEDUP

(57) Abstract

A scheduling and arbitration scheme in an input–buffered switch with speedup deterministic bandwidth and delay performance independent of switch size is presented. Within the framework of a crossbar architecture having a plurality of input channels and output channels, the scheduling and arbitration scheme determines the sequence of fixed–size packet or cell transmissions between the input channels and output channels satisfying the constraint that only one cell can leave an input channel and enter an output channel per phase in such a way that the arbitration delay is bounded for each cell awaiting transmission at the input channel. If the fixed–sized packets result from fragmentation of variable size packets, the scheduling and arbitration scheme implies that if delay guarantees are provided at the cell level, they are also provided to the initial variable size packets, re–assembled at the output channel, as well.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

METHOD FOR PROVIDING DELAYS INDEPENDENT OF SWITCH SIZE IN A CROSSBAR SWITCH WITH SPEEDUP

FIELD OF THE INVENTION

The present invention relates generally to variable and fixed size packet switches, and more particularly, to an apparatus and method for scheduling packet cell inputs through such packet switches.

BACKGROUND OF THE INVENTION

In the field of Integrated Services Networks, the importance of maintaining Quality of Service (QoS) for individual traffic streams (or flows) is generally recognized. Thus, such capability continues to be the subject of much research and development. Of particular interest is the delay experienced by an individual packet or cell. Good delay performance must be provided to all flows abiding to their service contract negotiated at connection setup, even in the presence of other potentially misbehaved flows. Many different methods have been developed to provide such performance in non-blocking switch architectures such as output buffered or shared memory switches. Several algorithms providing a wide range of delay guarantees for non-blocking architectures have been disclosed in the literature. See, for example, A. Parekh, "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks", MIT, Ph.D dissertation, June 1994; J. Bennett and H. Zhang, "WF2Q - Worst-case Fair Weighted Fair Queuing", Proc. IEEE INFOCOM'96; D. Stiliadis and A. Varma, "Frame-Based Fair Queuing: A New Traffic Scheduling Algorithm for Packet Switch Networks", Proc. IEEE INFOCOM '96; L. Zhang, "A New Architecture for Packet Switched Network Protocols," Massachusetts Institute of Technology, Ph.D Dissertation, July 1989; A. Charny, "Hierarchical Relative Error Scheduler: An Efficient Traffic Shaper for Packet Switching Networks," Proc. NOSSDAV '97, May 1997, pp. 283-294; and others. Schedulers capable of providing bandwidth and delay guarantees in non-blocking architectures are commonly referred to as "QoS-capable schedulers".

Typically, output-buffered or shared memory non-blocking architectures require the existence of high-speed memory. For example, an output-buffered switch requires that the speed of memory at each output must be equal to the total speed of all inputs. Unfortunately, the rate of the increase in memory speed available with current technology has not kept pace with the rapid growth in demand for providing large-scale integrated services networks. Because there

is a growing demand for large switches with total input capacity on the order of tens and hundreds of Gb/s, building an output buffered switch at this speed has become a daunting task given the present state of technology. Similar issues arise with shared memory switches as well.

As a result, many industrial and research architectures have adopted a more scalable approach, for example, crossbars. Details of such architectures may be had with reference to the following papers: T. Anderson, S. Owicki, J. Saxe, C. Thacker, "High Speed Switch Scheduling for Local Area Networks", Proc. Fifth Internat. Conf. on Architectural Support for Programming Languages and Operating Systems," Oct. 1992, pp. 98-110; and N. McKeown, M. Izzard, A. Mekkittikul, W. Ellersick and M. Horowitz, "The Tiny Tera: A Packet Switch Core." Even given the advances in the art, providing QoS in an input-queued crossbar switch remains a significant challenge.

A paper by N. McKeown, V. Anatharam and J. Warland, entitled "Achieving 100% Throughput in an Input-Queued Switch," Proc. IEEE INFOCOM '96, March 1996, pp. 296-302, describes several algorithms based on weighted maximum bipartite matching (defined therein) and capable of providing 100% throughput in an input-buffered switch. Unfortunately, the complexity of these algorithms is viewed as too high to be realistic for high-speed hardware implementations. In addition, the nature of the delay guarantees provided by these algorithms remains largely unknown.

Published research by D. Stiliadis and A. Varma, entitled "Providing Bandwidth Guarantees in an Input-Buffered Crossbar Switch," Proc. IEEE INFOCOM '95, April 1995, pp. 960-968, suggests that bandwidth guarantees in an input buffered crossbar switch may be realized using an algorithm referred to as Weighted Probabilistic Iterative Matching (WPIM), which is essentially a weighted version of the algorithm described in Anderson et al. Although the WPIM algorithm is more suitable for hardware implementations than that described by McKeown et al., it does not appear to provide bandwidth guarantees, and the delay performance in general has not been understood.

One method of providing bandwidth and delay guarantees in an input-buffered crossbar architecture uses statically computed schedule tables (an example of which is described in Anderson et al.); however, there are several significant limitations associated with this approach. First, the computation of schedule tables is extremely complex and time-consuming. Therefore, it can only be performed at connection setup-time. Adding a new flow or changing the rates of the existing flows is quite difficult and time-consuming, since such modification can require re-

computation of the whole table. Without such re-computation, it is frequently impossible to provide delay and even bandwidth guarantees even for a feasible rate assignment. Consequently, these table updates tend to be performed less frequently than may be desired. Second, there exists the necessity to constrain the supported rates to a rather coarse rate granularity and to restrict the smallest supported rate in order to limit the size of the schedule table. These limitations serve to substantially reduce the flexibility of providing QoS.

The search for scalable solutions which can provide high-quality QoS has led to several approaches. In one approach, an algorithm allows for the emulation of a non-blocking output-buffered switch with an output FIFO queue by using an input-buffered crossbar with speedup independent of the size of the switch. See B. Prabhakar and N. McKeown, "On the Speedup Required for Combined Input and Output Queued Switching," Computer Systems Lab. Technical Report CSL-TR-97-738, Stanford University. More specifically, this reference shows that such emulation is possible with a speedup of 4 and conjectures that a speedup of 2 may suffice. This result allows one to emulate a particular instantiation of a non-blocking output-buffered architecture without having to use the speedup of the order of the switch size, i.e., speedup equal to the number of ports. However, this algorithm is only capable of a very limited emulation of an output buffered switch with FIFO service. Furthermore, as described in the above-referenced technical report, such emulation does not provide any delay guarantees and the delay performance in general is not well understood. Its capability of providing bandwidth guarantees over a large time scale of a large time scale is limited to flows which are already shaped according to their rate at the input to the switch, and no bandwidth guarantees can be provided in the presence of misbehaved flows.

It should be noted that in speeded-up input buffered architectures the instantaneous rate of data entering an output channel may exceed the channel capacity. Therefore, buffering is required not only at the inputs, but also at the outputs. Therefore, input-buffered crossbar switches with speedup are also known as combined input/output buffered switches. Hereinafter, the more conventional term "speeded-up input-buffered crossbar" shall be used.

Another published study of speeded-up input buffered switches suggests that input-buffered switches with even small values of speedup may be capable of providing delays comparable to those of output-buffered switches, but is silent as to the dependence of these delays on the switch size within the framework described therein. See R. Guerin and K.

Sivarajan, "Delay and Throughput Performance of Speeded-up Input-Queuing Packet Switches," IBM Research Report RC 20892, June 1997.

Thus, there exists a present need in the art to provide adequate delay performance to guaranteed flows while utilizing the scalability of a crossbar architecture with speedup independent of switch size.

SUMMARY OF THE INVENTION

Accordingly, it is an object of the present invention to provide per-packet delays independent of the switch size in an input-buffered switch.

It is yet another object of the present invention to ensure protection to all flows against misbehaved flows.

It is still yet another object of the present invention to accommodate dynamically changing load and flow composition while operating at high speed, as well as avoid the imposition of artificial restrictions on supported rates.

In accordance with the purposes of the present invention, as embodied and described herein, the above and other purposes are attained by a method of providing delay performance independent of a switch size in an input-buffered switch with a speedup S greater than two having input channels and output channels for transferring cells therebetween. The method includes providing, to each of the input channels, per-output-channel queues to buffer cells awaiting transfer to the output channels, each per-output-channel queue being associated with a respective input channel and output channel, and having an assigned rate and an ideal service associated therewith; providing an arbiter to control transmission of buffered cells from input channels to output channels, the arbiter having a rate controller to schedule at a given cell slot the queues in the input channels, the rate controller to guarantee to each queue an amount of actual service that is within fixed bounds from the ideal service of the queue, the fixed bounds each being equal to one cell. For each per-output-channel queue, a pair of state variables is maintained including a first and a second state variable, the first state variable corresponding to an ideal beginning time of a next cell of the per-output queue and the second state variable corresponding to an ideal finishing time of transmission of the next cell of the per-output queue. The method further includes initializing the first and second state variables, the first state variable being equal to one and the second state variable being equal to one divided by the assigned rate; initializing an arbiter clock counter to count switch phases to zero; providing a set_match set and a set

_queues set; initializing the set_match set to include an empty set and the set_queues set to include all of the pairs; running the rate controller to select from the set_queues set ones of the pairs having a smallest eligible finish time first and, for the selected pair, updating the first state variable with the ideal finish time and the second state variable with an ideal beginning time plus one divided by the assigned rate; adding the selected pair to the set_match set; removing from the set_queues set those pairs corresponding to the same input channel and output channel as the selected pair; determining whether or not the set_queues set is empty. When the set_queues set is determined to be empty, the method includes notifying the input channels of the per-output-channel queues corresponding to those pairs added to the set_match set, incrementing the counter by one and returning to the step of initializing the set_match and set_queues sets; and when the set_queues set is determined to be not empty, then returning to the step of running the rate controller.

The present invention achieves several important goals. It provides per cell/packet delay independent of the switch size comparable to delay guarantees associated with non-blocking output-buffered architectures, while utilizing the scalability of a crossbar. It allows arbitrary assignment of rates (as long as the rates are feasible in the sense that the sum of all rates does not exceed the total available bandwidth at any input or any output). Additionally, it allows the flexibility to quickly admit new flows and change the rate assignment of existing flows. Moreover, it ensures protection of well-behaved flows against misbehaved flows.

More specifically, simulations indicate that such a system is capable of providing delays comparable to those of an output buffered switch, with any speedup of greater than or equal to two, and the delays observed are independent of switch size.

While the invention is primarily related to providing per-packet/cell delays to guaranteed flows, it can be used in conjunction with best-effort traffic as well. If best effort traffic is present, it is assumed that the invention as described herein is run at an absolute priority over any scheduling algorithm for best effort traffic.

BRIEF DESCRIPTION OF THE DRAWINGS

The above objects, features and advantages of the present invention will become more apparent from the following description of the embodiments of the present invention illustrated in the accompanying drawings, wherein:

FIG. 1 is block diagram depicting an input-buffered crossbar switch capable of utilizing per-output-channel queue scheduling and arbitration schemes in accordance with the present invention; and

FIG. 2 is a flow diagram illustrating a queue scheduling and arbitration scheme for providing delays independent of the switch size in accordance with the present invention.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

Referring to FIG. 1, with like reference numerals identifying like elements, there is shown an input-buffered crossbar switch 10 implementing a crossbar arbitration scheme in accordance with the present invention. As illustrated in FIG. 1, the underlying architecture of the input-buffered crossbar switch 10 is represented as an n x m crossbar. Here, "n" is the number of input channels i ($1 \leq i \leq n$) 12 and "m" is the number of output channels j ($1 \leq j \leq m$) 14. Each input channel has one or more input ports 16, each of which corresponds to a physical input link 18. Similarly, the output channels each have one or more output ports 20, each corresponding to a physical output link 22. The input channels 12 are connected to the output channels 14 by way of a crossbar unit 24. It will be understood by those skilled in the art that the crossbar unit as depicted in FIG. 1 includes a crossbar switch fabric of known construction, the details of which have been omitted for purposes of simplification. It is the crossbar switch fabric that is responsible for transferring cells between input and output channels.

In the embodiment shown, the total capacity of all input channels and all output channels is assumed to be the same, although the capacity of individual links may be different. Hereinafter, the capacity of a single channel is denoted by r_c. The speed of the switch fabric, denoted by r_sw, is assumed to be S times faster than the speed of any channel. In general, the switch and the channel clocks are not assumed to be synchronized. The speedup values may be arbitrary (and not necessarily integer) values in the range of $1 \leq S \leq n$. It is further assumed that the switch operates in phases of duration T_sw defined as the time needed to transmit a unit of data at speed r_sw. Such phases are referred to as matching phases. In this disclosure, a unit of data shall be referred to as a *cell*. Accordingly, a switch can move at most one cell from each input channel and at most one cell to each output channel at each matching phase. Therefore, on average, a switch with speedup S can move S cells from each input channel and S cells to each output channel. At $S=n$, the switch is equivalent to the output buffered switch.

Although not shown in FIG. 1, packets received on a given input link 18 are typically buffered at the input ports. Also, each flow to which the received packets correspond may be allocated a separate buffer or queue at the input channel. These "per-flow" queues may be located in an area of central memory within the input channel. Alternatively, flow queues may be located in a memory in the input ports associated with the input channel. When the packets received from the input links are of variable length, they are fragmented into fixed-size cells. If the packets arriving at the switch all have a fixed length, e.g., a cell in ATM networks, no fragmentation is required. In packet switching networks, where arriving packets are of different sizes, the implementation is free to choose the size of the cell as is convenient. The tradeoff in the choice of this size is that the smaller the cell, the better delays can be provided, but the faster the arbitration must be (and therefore the more expensive the switch). In addition, small cell size causes larger fragmentation overhead. Upon arrival and after possible fragmentation, cells are mapped to a corresponding flow (based on various classifiers: source address, destination address, protocol type, etc.). Once mapped, the cells are placed in the appropriate "per-flow" queue.

Associated with each guaranteed flow is some rate r_f, which is typically established at connection setup time (e.g., via RSVP). Rates assigned to guaranteed flows can also be changed during a renegotiation of service parameters as allowed by the current RSVP specification. It is assumed that the rate assignment is feasible, i.e., the sum of the rates of all flows at each input port channel does not exceed the capacity of this input port channel, and the sum of rates of all flows across all input ports destined to a particular output port does not exceed the capacity of that output port. If the sum of port capacities equals the channel capacity as assumed here, the feasibility of rates across all input and output ports implies the feasibility of rates across all input and output channels. Included in the rate r_f assigned to the flow is any overhead associated with packet fragmentation and re-assembly. The actual data rate negotiated at connection setup may therefore be lower. For networks with fixed packet size, such as ATM, however, no segmentation and re-assembly is required. Thus, no overhead is present.

As shown in FIG. 1, each input channel i 12 has m virtual output queues (VOQs) or per-output-channel queues 26 (also referred to as per-output or virtual output queues), denoted by $Q(i,j)$, $1 \leq j \leq m$, one for each output channel j 14. In the embodiment shown in FIG. 1, the input channel maintains a single flow-level scheduler $S_f(i)$ 28, which needs to schedule only a single flow per cell time. Once scheduler $S_f(i)$ schedules some flow f, it adds the index of this

flow *f* (or, alternatively, the head of the line (HOL) cell of flow *f*) to the tail of queue *Q(i,j)*. Thus, depending on the implementation, *Q(i,j)* may contain either cells or pointers to cells of individual flows. Any known QoS-capable scheduler, such as those described above, can be used for the *S_f* scheduler.

In another variation, each input channel could maintain one flow-level scheduler *S_f(i,j)* for each output. When the input channel *i* needs to transmit a cell to a given output *j*, it invokes scheduler *S_f(i,j)* to determine which flow destined to *j* should be chosen. Unlike the option described above, in which scheduler *S_f(i)* can run at link speed, the flow-level schedulers *S_f(i,j)* must be capable of choosing up to S cells per cell time as it is possible that this input may need to send a cell to the same output in all S matching phases of the current cell slot. In yet another approach, the input can run m parallel *S_f* schedulers, one per output. Each of these schedulers may schedule $1 \leq k \leq S$ cells per cell time. When a flow is scheduled by *S_f*, an index to this flow is added to *Q(i,j)*.

Also included in the input-buffered crossbar switch 10 is an arbiter 30 as shown in FIG. 1. It is the arbiter's responsibility to determine which of the input channels should be able to transmit a cell to particular output channels, i.e., cells from which per-output-channel queues should be transmitted. It is assumed that arbiter 24 operates in matching phases. The duration of each phase is equal to the duration of the channel cell slot divided by the speedup *S*. The goal of the arbiter is to compute a maximal (conflict-free) match between the input and output channels so that at most one cell leaves any input channel and at most one cell enters any output channel during a single matching phase. Although the term "maximal match" (or, alternatively, "maximal matching") is well understood by those skilled in the art, a definition may be had with reference to papers by N. McKeown et al. and Stiliadis et al., cited above, as well as U.S. Patent No. 5,517,495 to Lund et al.

As explained above, during each of its matching phases, the arbiter decides which input can send a cell to which output by computing a maximal matching between all inputs and all outputs. The algorithm used to compute the maximal match is described in detail in paragraphs to follow. Once the matching is completed, the arbiter notifies each input of the output to which it can send a cell by sending to the input channel the index of the per-output queue from which the cell is to be transmitted. The input channel then picks a cell to send to that output channel and the cell is transmitted to the output channel. As shown in FIG. 1, the arbiter 30 maintains

for each input/output pair i,j, a pair of variables (b_i,j, f_i,j) denoted as $A(i,j)$ 32. How the arbiter utilizes these input/output pairs will be described in detail later with reference to FIG. 2.

When an input channel 12 receives from the arbiter 24 the index of the $Q(i,j)$ corresponding to the output channel 14 for the current matching phase, it forwards the HOL cell of $Q(i,j)$ (or, alternatively, the cell pointed to by the HOL pointer in $Q(i,j)$) to the output channel j. If $Q(i,j)$ is empty that is, there is no cell of a guaranteed flow in the queue, then a cell of a lower-priority service destined to the same output is sent instead. If there is no best effort traffic at this input matching phase, then no cell is sent.

Although not shown in FIG. 1, a cell forwarded by an input channel i to an output channel j is added to a queue maintained by the output channel. A variety of queuing disciplines can be used, such as FIFO, per-input-port, or per flow. If the queue is not a simple FIFO, each output has an additional scheduler, shown in FIG. 1 as output scheduler S_o 34. This output scheduler determines the order in which cells are transmitted onto the output link from the output channel. It is assumed that any required reassembly occurs before S_o is used, so that S_o schedules packets rather than cells.

Any known QoS-capable scheduler such as those mentioned above can be used for the schedule S_o.

Since each scheduler S_f, S_o operates independently of the other, the delay of an individual cell in the switch is the sum of the delay of this cell under its input and output schedulers S_f and S_o, plus the delay due to the potential arbitration conflicts. The delay of a packet segmented in cells is comprised of the delay experienced by its last cell plus the segmentation and re-assembly delays.

Still referring to FIG. 1, it can now be appreciated that, with respect to each input channel, each of the queues Q(i,j) contains cells (or pointers to cells) which have already been scheduled by S_f but which have not yet been transmitted to their destination output channel with which the VOQ is associated due to arbitration conflicts. The present invention undertakes the task of determining the sequence of transmissions between input channels and output channels satisfying the crossbar constraint that only one cell can leave an input channel and enter an output channel per phase in such a way that the arbitration delay is bounded for each cell awaiting its transmission at the input channel.

Now referring to FIG. 2, there is illustrated the actions of the arbiter with respect to scheduling the per-output-channel queues 40 in accordance with the present invention. As

previously indicated, the arbiter maintains a pair of variables (b_i,j, f_i,j) or $A(i,j)$ for each input/output pair i,j. These variables b_i,j and f_i,j will be referred to as starting time and finish time, respectively. The starting time is the ideal beginning time of transmission of the next cell of the queue with which the input/output pair is associated. The finish time is the ideal finishing time of transmission of the next cell of the queue with which the input/output pair is associated. At initial step 42, the arbiter obtains for each input/output pair i,j the rate r_i,j, which is the sum of the assigned rates of all flows going from input i to output j. Also, in the same step, variables b_i,j and f_i,j are initialized (to zero and $1/r_i,j$, respectively) and a count value *time* is set to zero.

As further illustrated in FIG. 2, at each matching phase the arbiter computes the maximal match as follows. In step 44, the arbiter initializes a *Set_Match* set to an empty set and a *Set_Queues* set to all $A(i,j)$. Now referring to step 46 in FIG. 2, the arbiter selects the pair $A(i,j)$ having the smallest finish time f_i,j among all eligible pairs, where eligible pairs are defined as those whose starting time b_i,j is at or before the current time. In step 48, the arbiter adds the pair selected in step 46 to *Set_Match*, updates the variables such that $b_i,j=f_i,j$ and $f_i,j=f_j,j+1/r_i,j$ as indicated in step 50 and, in step 52, removes from set *Set_Queues* all pairs corresponding to the input and/or output of the A(i,j) selected in step 46. If there are any pairs remaining in *Set_Queues* (step 54), the arbiter returns to step 46 and performs the next iteration of the matching process. Otherwise, the matching is complete. In step 56, for each $A(i,j)$ in the match, the arbiter informs the input i to send to all output j. As can be seen, the $A(i,j)$ in the match correspond to the per-output-channel queues Q(i,j) from which a cell should be transmitted in the current matching phase. The arbiter then proceeds to the next matching phase, incrementing count *time* by one (step 58) and updating the rates r_i,j as necessary (step 60) before returning to step 44.

In an alternative input-buffered switch algorithm described in a co-pending application which runs a separate version of the rate controller per input and performs arbitration using the scheduling times of the rate controllers, the delay bound is a function of the size of the switch (i.e., the number of input/channels). In contrast, the arbiter of the present invention runs a single rate controller across all queues regardless of the input or output channels to which they correspond and uses finish times (rather than scheduled times) as described above. Also, in the above-referenced co-pending application, the input rate controllers which schedule per-output queues at each input are oblivious to potential arbitration conflicts. The arbitration conflicts are resolved at the arbiter using timestamps of the scheduling times of the input rate controllers.

Here, in the present invention, the rate controller which is run in the arbiter uses ideal start and finish times of all input/output pairs directly and explicitly resolves arbitration conflicts as part of the operation of the rate controller. Hence, the advantage of the present invention is that the observed delays are independent of the size of the switch and depends only on the rate of the flow. However, in the present invention the rate controller must operate at the faster speed of the switch fabric whereas the input channel rate-controllers in the co-pending application need to operate at a slower channel speed. Likewise, the size of the input to the rate-controller in the present invention is nxm, whereas in the co-pending invention the input to each of the rate-controllers is only m. As a result, the implementation of the co-pending invention may be less expensive, especially at high speeds.

While the disclosed input-buffered switch and scheduling method has been particularly shown and described with reference to the preferred embodiments, it will be understood by those skilled in the art that various modifications in form and detail may be made therein without departing from the scope and spirit of the invention as set forth by the claims. Accordingly, modifications such as those suggested above, but not limited thereto, are to be considered within the scope of the claims.

CLAIMS

1. A method of providing delay performance independent of a switch size in an input-buffered switch with a speedup S greater than two having input channels and output channels for transferring cells therebetween, the method comprising:

providing, to each of the input channels, per-output-channel queues to buffer cells awaiting transfer to the output channels, each per-output-channel queue being associated with a respective input channel and output channel, and having an assigned rate and an ideal service associated therewith;

providing an arbiter to control transmission of buffered cells from input channels to output channels, the arbiter having a rate controller to schedule at a given cell slot the queues in the input channels, the rate controller to guarantee to each queue an amount of actual service that is within fixed bounds from the ideal service of the queue, the fixed bounds each being equal to one cell;

for each per-output-channel queue, maintaining a pair of state variables including a first and a second state variable, the first state variable corresponding to an ideal beginning time of a next cell of the per-output queue and the second state variable corresponding to an ideal finishing time of transmission of the next cell of the per-output queue;

initializing the first and second state variables, the first state variable being equal to one and the second state variable being equal to one divided by the assigned rate;

initializing an arbiter clock counter to count switch phases to zero;

providing a set_match set and a set _queues set;

initializing the set_match set to include an empty set and the set_queues set to include all of the pairs;

running the rate controller to select from the set_queues set ones of the pairs having a smallest eligible finish time first and, for the selected pair, updating the first state variable with the ideal finish time and the second state variable with an ideal beginning time plus one divided by the assigned rate;

adding the selected pair to the set_match set;

removing from the set_queues set those pairs corresponding to the same input channel and output channel as the selected pair;

determining whether or not the set_queues set is empty;

when the set_queues set is determined to be empty, notifying the input channels of the per-output-channel queues corresponding to those pairs added to the set_match set, incrementing the counter by one and returning to the step of initializing the set_match and set_queues sets; and

when the set_queues set is determined to be not empty, then returning to the step of running the rate controller.

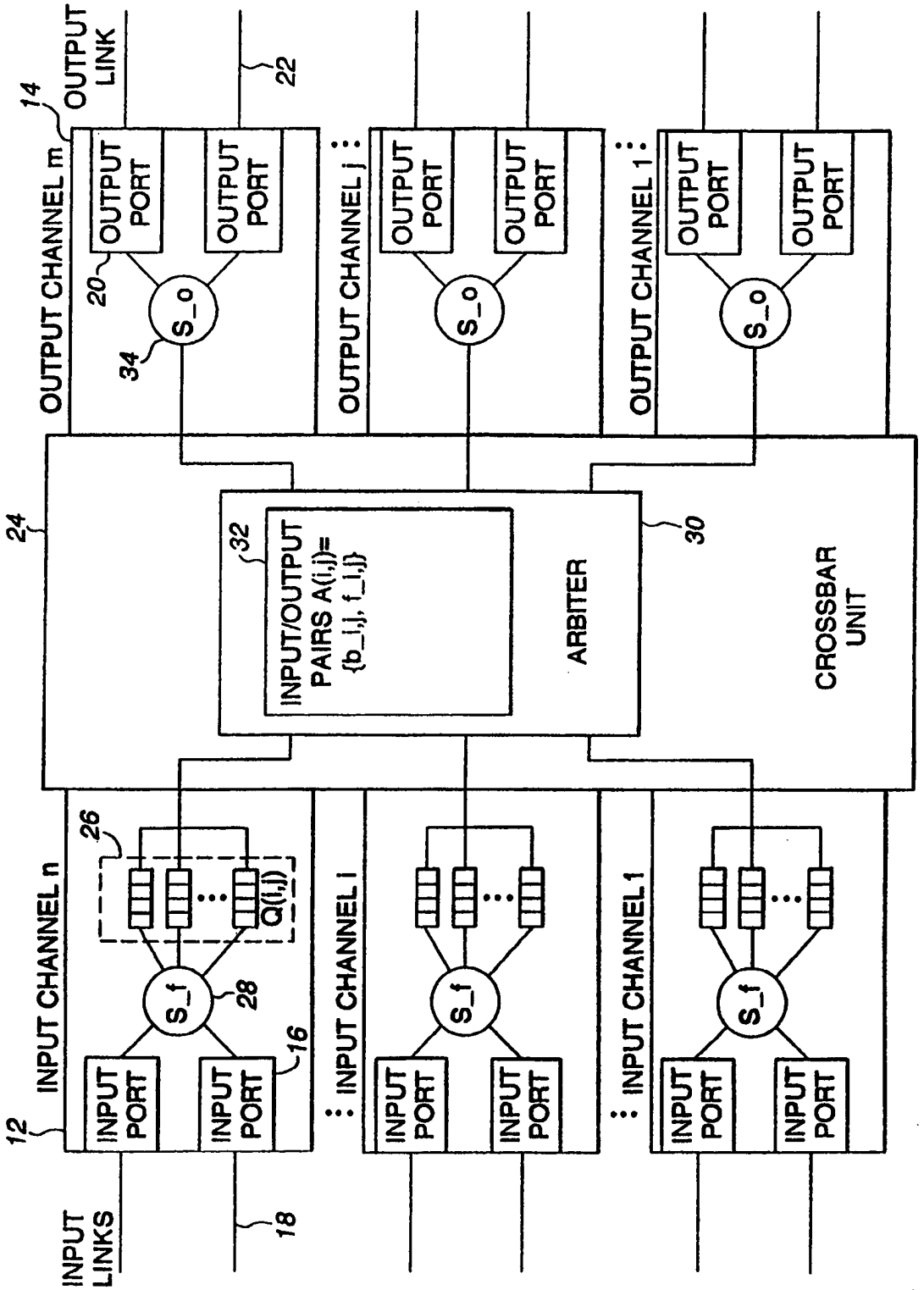

OUTPUT LINK
14
22
OUTPUT CHANNEL m
OUTPUT CHANNEL j
OUTPUT CHANNEL 1
OUTPUT PORT
20
34
S_o
24
32
INPUT/OUTPUT PAIRS A(i,j)= {b_i,j, f_i,j}
ARBITER
30
CROSSBAR UNIT
26
Q(i,j)
INPUT CHANNEL n
INPUT CHANNEL I
INPUT CHANNEL 1
S_f
28
16
INPUT PORT
12
INPUT LINKS
18
10

FIG. 1

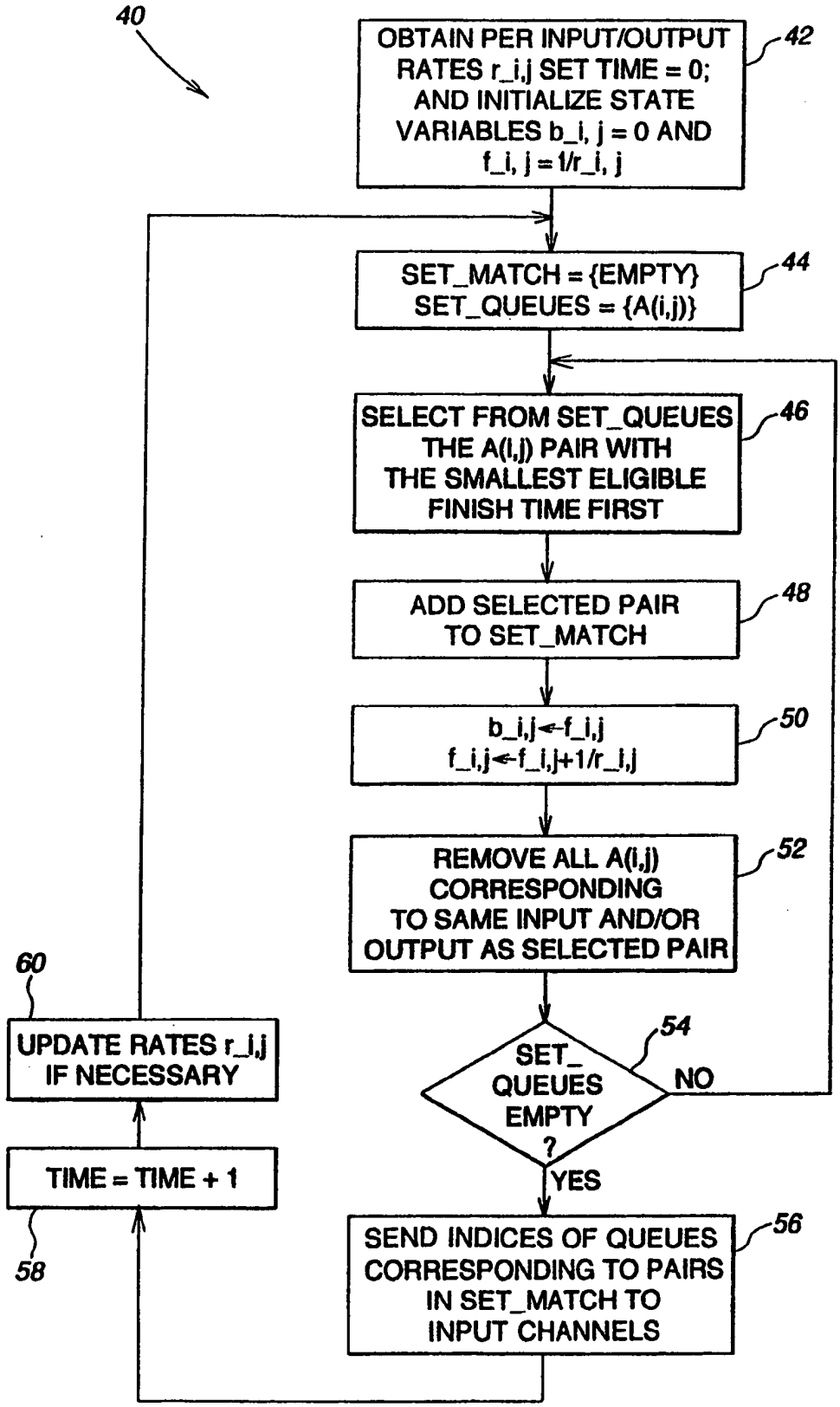

40
OBTAIN PER INPUT/OUTPUT RATES r_i,j SET TIME = 0; AND INITIALIZE STATE VARIABLES b_i, j = 0 AND f_i, j = 1/r_i, j
42
SET_MATCH = {EMPTY}
SET_QUEUES = {A(i,j)}
44
SELECT FROM SET_QUEUES THE A(I,J) PAIR WITH THE SMALLEST ELIGIBLE FINISH TIME FIRST
46
ADD SELECTED PAIR TO SET_MATCH
48
b_i,j←f_i,j
f_i,j←f_i,j+1/r_i,j
50
REMOVE ALL A(i,j) CORRESPONDING TO SAME INPUT AND/OR OUTPUT AS SELECTED PAIR
52
SET_ QUEUES EMPTY ?
54
NO
YES
SEND INDICES OF QUEUES CORRESPONDING TO PAIRS IN SET_MATCH TO INPUT CHANNELS
56
TIME = TIME + 1
58
UPDATE RATES r_i,j IF NECESSARY
60

FIG. 2

INTERNATIONAL SEARCH REPORT

national Application No
PCT/US 99/00684

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 H04L12/56 H04Q11/04

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 H04L H04Q

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	GB 2 293 720 A (ROKE MANOR RESEARCH) 3 April 1996 see page 4, line 10 - page 6, line 24	1
A	EP 0 817 436 A (NEWBRIDGE NETWORKS CORP ;XEROX CORP (US)) 7 January 1998 see page 6, line 49 - page 8, line 53	1
A	WO 97 31460 A (HAL COMPUTER SYSTEMS INC) 28 August 1997 see page 5, line 26 - page 6, line 25	1

☐ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
18 May 1999	27/05/1999

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Lindner, A

3

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Information on patent family members

International Application No
PCT/US 99/00684

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
GB 2293720 A	03-04-1996	CA 2158324 A	31-03-1996
		EP 0705007 A	03-04-1996
		JP 8125669 A	17-05-1996
		US 5734650 A	31-03-1998
EP 0817436 A	07-01-1998	EP 0817431 A	07-01-1998
		EP 0817432 A	07-01-1998
		EP 0817433 A	07-01-1998
		EP 0817434 A	07-01-1998
		EP 0817435 A	07-01-1998
		JP 10242999 A	11-09-1998
		JP 10190691 A	21-07-1998
		JP 10190692 A	21-07-1998
WO 9731460 A	28-08-1997	US 5892766 A	06-04-1999
		EP 0823167 A	11-02-1998